UNCLASSIFIED

AD

AD-E403 690

Technical Report ARWSE-TR-14027

WAITING ON MORE THAN 64 HANDLES

Tom Nealis

October 2015

U.S. ARMY ARMAMENT RESEARCH, DEVELOPMENT AND ENGINEERING CENTER

Weapons and Software Engineering Center

Picatinny Arsenal, New Jersey

UNCLASSIFIED

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-01-0188

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (*From - To*)
October 2015	Final	

4. TITLE AND SUBTITLE

WAITING ON MORE THAN 64 HANDLES

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHORS

Tom Nealis

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
U.S. Army ARDEC, WSEC
Fire Control Systems & Technology Directorate
(RDAR-WSF-M)
Picatinny Arsenal, NJ 07806-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Army ARDEC, ESIC
Knowledge & Process Management (RDAR-EIK)
Picatinny Arsenal, NJ 07806-5000

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
Technical Report ARWSE-TR-14027

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

It's often necessary for a Windows based program to wait on a handle to a thread or some other resource during its execution. Occasionally, it's necessary for a program to wait on many handles at one time. If the number of handles exceeds MAXIMUM_WAIT_OBJECTS, which is currently defined as 64, then some extra code must be written to deal with this.

15. SUBJECT TERMS

Handle MAXIMUM_WAIT_OBJECTS WaitForMultipleObjects

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
					Tom Nealis
a. REPORT	b. ABSTRACT	c. THIS PAGE			19b. TELEPHONE NUMBER (Include area code)
U	U	U	SAR	7	(973) 724-8048

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

CONTENTS

INTRODUCTION

The need to wait on more than 64 handles was first encountered while upgrading some old software to run more efficiently by breaking down tasks to run in multiple worker threads. Almost 200 threads were spawned and then it was found that there was a maximum number of handles that could be waited on. The maximum was set by a global definition of MAXIMUM_WAIT_OBJECTS, which equals 64. All of the threads/tasks needed to be completed first in order to report the completion of the process to the calling function.

METHODOLOGY

The initial situation presented was having all of the handles to the threads inside a std::vector. The general idea to handle this situation was to break down the number of thread handles into 64 or less chunks and spawn a subsequent thread for each group. That thread would then wait on the 64 thread handles of that group. Then, one would wait on all the handles produced by spawning a thread for each group.

The following sequence is the block of code/algorithm that was created for the situation of having more than 64 handles:

```
std::vector<HANDLE> new_handles;
int threads_needed = FCT_Threads.size() / MAXIMUM_WAIT_OBJECTS;
threads_needed += ((FCT_Threads.size() % MAXIMUM_WAIT_OBJECTS) == 0) ? 0 : 1;
for(int i = 0; i < threads_needed; ++i)
{
    WaitThreadData* data = new WaitThreadData;
    if((i + 1) < threads_needed)
    {
      data->quantity = MAXIMUM_WAIT_OBJECTS;
      memcpy_s(data->handles, sizeof(HANDLE) * MAXIMUM_WAIT_OBJECTS, &FCT_Threads[i * MAXIMUM_WAIT_OBJECTS],
sizeof(HANDLE) * MAXIMUM_WAIT_OBJECTS);
    }
    else
    {
      //partial group
      data->quantity = FCT_Threads.size() % MAXIMUM_WAIT_OBJECTS;
      memcpy_s(data->handles, sizeof(HANDLE) * data->quantity, &FCT_Threads[i * MAXIMUM_WAIT_OBJECTS], sizeof(HANDLE) * data-
>quantity);
    }

    HANDLE ct = CreateThread(NULL, 0, CFCT_GenDlg::WaitThreads, (LPVOID)data, 0, NULL);
      new_handles.push_back(ct);
}

unsigned long ret = WaitForMultipleObjects(new_handles.size(), new_handles.data(), true, INFINITE);
```

The first step the code takes is to figure out how many threads need to spawn. This is achieved by dividing by the maximum wait objects and then checking for a remainder. If there is a remainder, then an additional thread will be needed for the partial group. The next step is to iterate once for the number of threads needed, filling the parameter struct appropriately. The parameter stuct is comprised of an array of handles with a maximum size of MAX_WAIT_OBJECTS and an integer holding the total quantity.

Once all the “waitthreads” are created, a final wait is initiated on the current thread with the handles of all the created “waitthreads.” Upon return, all the threads would have completed running their tasks and can be cleaned up.

CONCLUSIONS

This code snippet is not the silver bullet for all problems associated with waiting on more than 64 handles. It will, however, work for up to 4,096 generated threads. It does give a very good starting point for anyone dealing with a similar issue.

DISTRIBUTION LIST

U.S. Army ARDEC
ATTN: RDAR-EIK
RDAR-WSF-M, T. Nealis
Picatinny Arsenal, NJ 07806-5000

Defense Technical Information Center (DTIC)
ATTN: Accessions Division
8725 John J. Kingman Road, Ste 0944
Fort Belvoir, VA 22060-6218

GIDEP Operations Center
P.O. Box 8000
Corona, CA 91718-8000
gidep@gidep.org

REVIEW AND APPROVAL OF ARDEC TECHNICAL REPORTS

Waiting on more than 64 handles
Title

Date received by LCSD

Thomas M. Nealis
Author/Project Engineer

Report number (to be assigned by LCSD)

X8048 31
Extension Building

RDAR-WSF-M
Author's/Project Engineers Office
(Division, Laboratory, Symbol)

PART 1. **Must be signed before the report can be edited.**

a. The draft copy of this report has been reviewed for technical accuracy and is approved for editing.

b. Use Distribution Statement A_X___, B__, C___, D____, E____, F____ or X____ for the reason checked on the continuation of this form. Reason: ____Operational Use__________

1. If Statement A is selected, the report will be released to the National Technical Information Service (NTIS) for sale to the general public. Only unclassified reports whose distribution is not limited or controlled in any way are released to NTIS.

2. If Statement B, C, D, E, F, or X is selected, the report will be released to the Defense Technical Information Center (DTIC) which will limit distribution according to the conditions indicated in the statement.

c. The distribution list for this report has been reviewed for accuracy and completeness.

Patricia Alameda 9/9/15
Division Chief (Date)

PART 2. To be signed either when draft report is submitted or after review of reproduction copy.

This report is approved for publication.

Patricia Alameda 9/9/15
Division Chief (Date)

Andrew Pskowski 10/16/15
RDAR-CIS (Date)

LCSD 49 supersedes SMCAR Form 49, 20 Dec 06